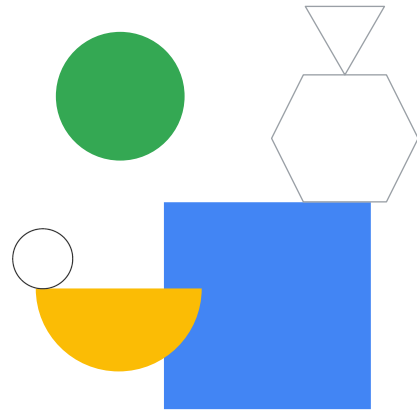


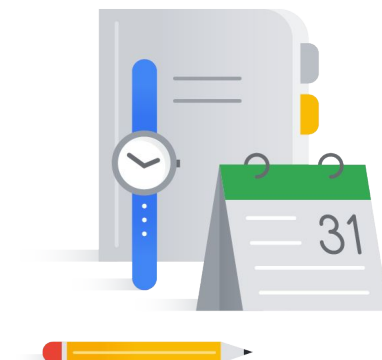
Resource Monitoring



In this module, I'll give you an overview of the resource monitoring options in Google Cloud.

Agenda

- 01 Google Cloud Observability
- 02 Monitoring
 - Lab: Resource Monitoring
- 03 Logging
- 04 Error Reporting
- 05 Tracing
- 06 Profiling
- 07 Partner Integrations



The features covered in this module rely on Google Cloud Observability, a service that provides monitoring, logging, and diagnostics for your applications.

In this module, we are going to explore the Cloud Monitoring, Cloud Logging, Error Reporting, Cloud Trace, and Cloud Profiler services.

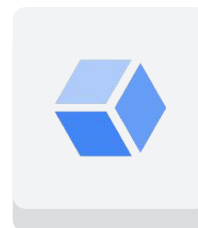


Google Cloud Observability

Let me start by giving you a high-level overview of Google Cloud Observability and its features.

Google Cloud Observability overview

- Integrated monitoring, logging, diagnostics
- Manages across platforms
 - Google Cloud and AWS
 - Dynamic discovery of Google Cloud with smart defaults
 - Open-source agents and integrations
- Access to powerful data and analytics tools
- Collaboration with third-party software

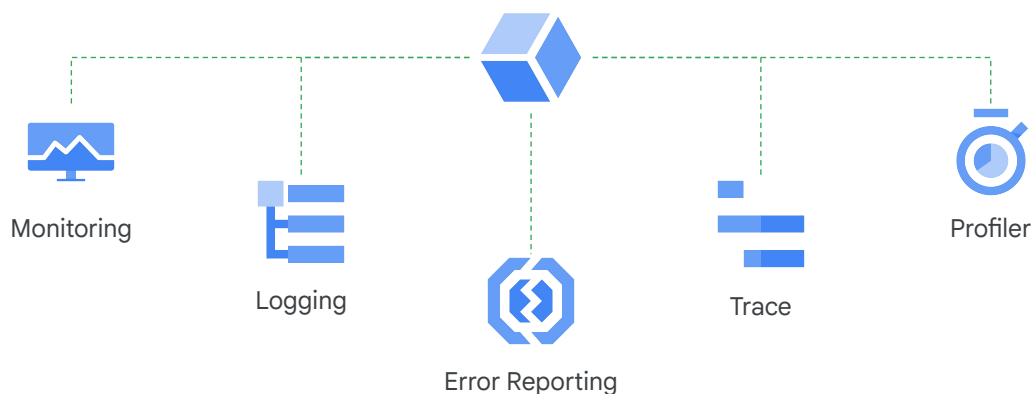


Google Cloud
Observability

Google Cloud Observability dynamically discovers cloud resources and application services based on deep integration with Google Cloud and Amazon Web Services. Because of its smart defaults, you can have core visibility into your cloud platform in minutes.

This provides you with access to powerful data and analytics tools plus collaboration with many different third-party software providers.

Multiple integrated products



As mentioned earlier, Google Cloud Observability has services for monitoring, logging, error reporting, and fault tracing. You only pay for what you use, and there are free usage allotments so that you can get started with no upfront fees or commitments. For more information about pricing, refer to the link in the Course Resources.

Now, in most other environments, these services are handled by completely different packages, or by a loosely integrated collection of software. When you see these functions working together in a single, comprehensive, and integrated service, you'll realize how important that is to creating reliable, stable, and maintainable applications.

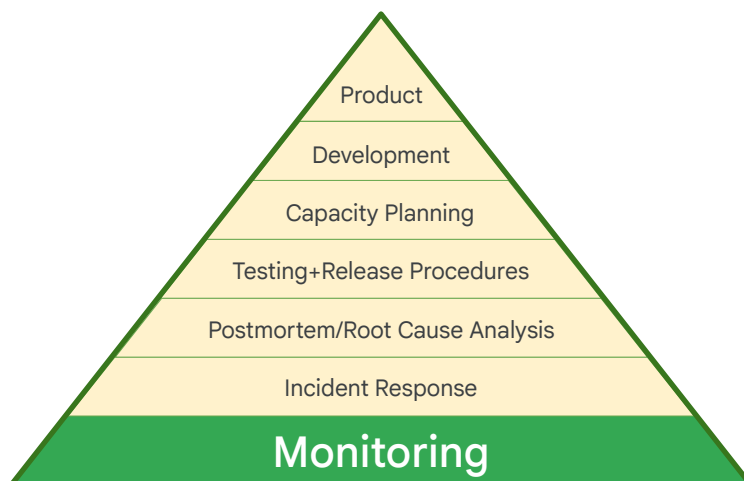
[\[https://cloud.google.com/stackdriver/pricing\]](https://cloud.google.com/stackdriver/pricing)



Monitoring

Now that you understand Google Cloud Observability from a high-level perspective, let's look at Cloud Monitoring.

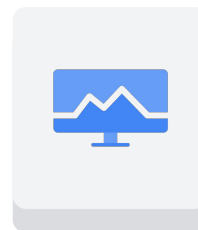
Site reliability engineering



Monitoring is important to Google because it is at the base of site reliability engineering, a discipline that applies aspects of software engineering to operations whose goals are to create ultra-scalable and highly reliable software systems. This discipline has enabled Google to build, deploy, monitor, and maintain some of the largest software systems in the world.

Monitoring

- Dynamic config and intelligent defaults
- Platform, system, and application metrics
 - Ingests data: Metrics, events, metadata
 - Generates insights through dashboards, charts, alerts
- Uptime/health checks
- Dashboards
- Alerts



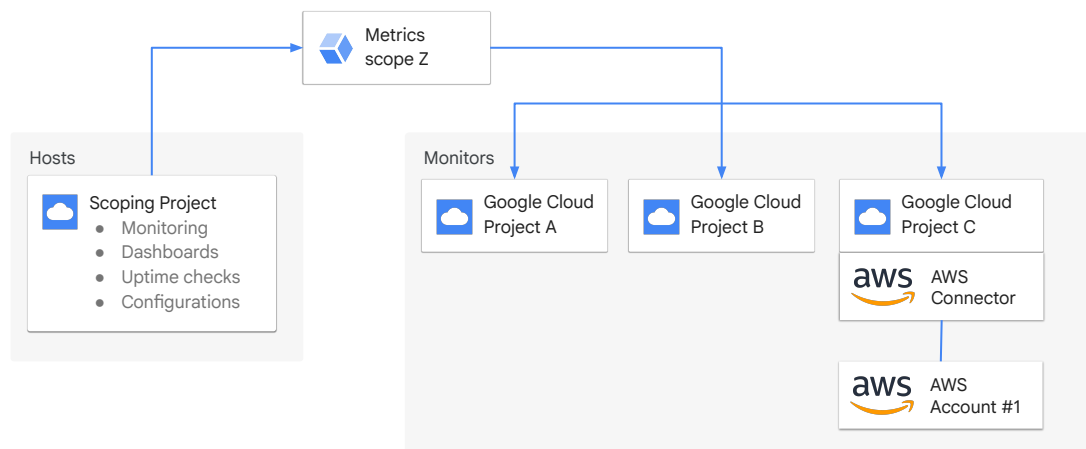
Monitoring

Cloud Monitoring dynamically configures monitoring after resources are deployed and has intelligent defaults that allow you to easily create charts for basic monitoring activities.

This allows you to monitor your platform, system, and application metrics by ingesting data, such as metrics, events, and metadata. You can then generate insights from this data through dashboards, charts, and alerts.

For example, you can configure and measure uptime and health checks that send alerts via email.

A metrics scope is the root entity that holds monitoring and configuration information



Google Cloud

A metrics scope is the root entity that holds monitoring and configuration information in Cloud Monitoring. Each metrics scope can have between 1 and 375 monitored projects. Monitoring data for all projects in that scope will be visible.

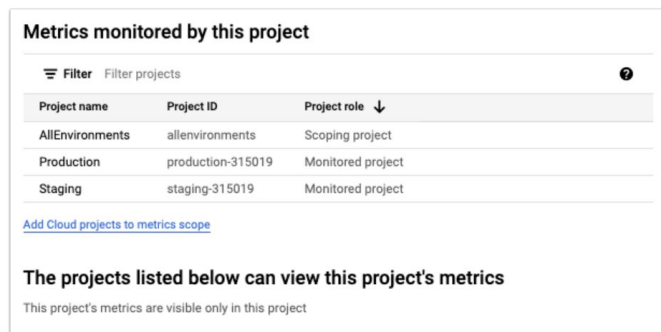
A metrics scope contains the custom dashboards, alerting policies, uptime checks, notification channels, and group definitions that you use with your monitored projects. A metrics scope can access metric data from its monitored projects, but the metrics data and log entries remain in the individual projects.

The first monitored Google Cloud project in a metrics scope is called the scoping project, and it must be specified when you create the metrics scope. The name of that project becomes the name of your metrics scope. To access an AWS account, you must configure a project in Google Cloud to hold the AWS Connector.

[<https://cloud.google.com/monitoring/settings#concept-scope>]

A metrics scope is a “single pane of glass”

- Determine your monitoring needs up front.
- Consider using separate metrics scopes for data and control isolation.



The screenshot shows a web interface titled "Metrics monitored by this project". It includes a filter bar with a "Filter" button and a "Filter projects" label. Below this is a table with three columns: "Project name", "Project ID", and "Project role" with a downward arrow. The table lists three projects: "AllEnvironments" (allenvironments, Scoping project), "Production" (production-315019, Monitored project), and "Staging" (staging-315019, Monitored project). A link "Add Cloud projects to metrics scope" is below the table. At the bottom, a section titled "The projects listed below can view this project's metrics" includes a note: "This project's metrics are visible only in this project".

Project name	Project ID	Project role ↓
AllEnvironments	allenvironments	Scoping project
Production	production-315019	Monitored project
Staging	staging-315019	Monitored project

[Add Cloud projects to metrics scope](#)

The projects listed below can view this project's metrics

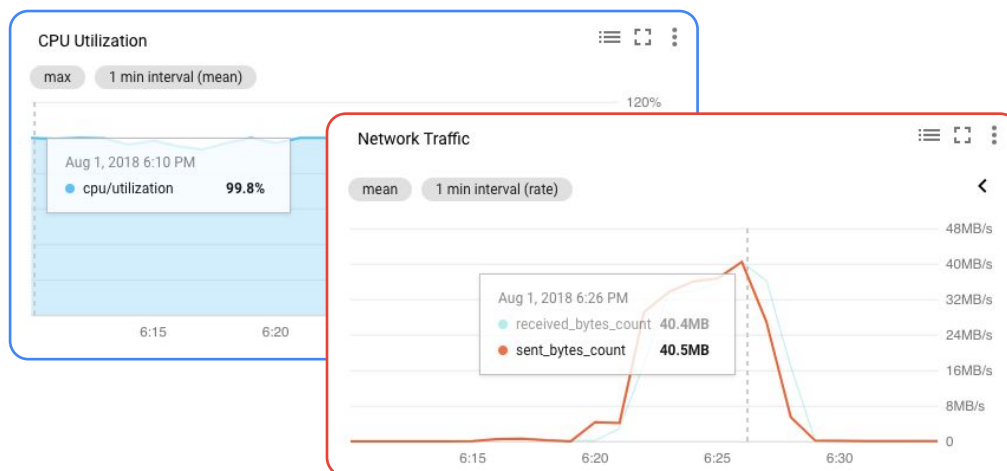
This project's metrics are visible only in this project

Because metrics scopes can monitor all your Google Cloud projects in a single place, a metrics scope is a “single pane of glass” through which you can view resources from multiple Google Cloud projects and AWS accounts. All users of Google Cloud Observability with access to that metrics scope have access to all data by default.

This means that a role assigned to one person on one project applies equally to all projects monitored by that metrics scope.

In order to give people different roles per project and to control visibility to data, consider placing the monitoring of those projects in separate metrics scopes.

Dashboards visualize utilization and network traffic



Google Cloud

Cloud Monitoring allows you to create custom dashboards that contain charts of the metrics that you want to monitor. For example, you can create charts that display your instances' CPU utilization, the packets or bytes sent and received by those instances, and the packets or bytes dropped by the firewall of those instances.

In other words, charts provide visibility into the utilization and network traffic of your VM instances, as shown on this slide. These charts can be customized with filters to remove noise, groups to reduce the number of time series, and aggregates to group multiple time series together.

For a full list of supported metrics, please refer to the documentation.

[\[https://cloud.google.com/monitoring/api/metrics_gcp\]](https://cloud.google.com/monitoring/api/metrics_gcp)

Alerting policies can notify you of certain conditions



Google Cloud

Although charts are extremely useful, they can only provide insight while someone is looking at them. But what if your server goes down in the middle of the night or over the weekend? Do you expect someone to always look at dashboards to determine whether your servers are available or have enough capacity or bandwidth?

If not, you want to create alerting policies that notify you when specific conditions are met.

For example, as shown on this slide, you can create an alerting policy when the network egress of your VM instance goes above a certain threshold for a specific timeframe. When this condition is met, you or someone else can be automatically notified through email, SMS, or other channels in order to troubleshoot this issue.

You can also create an alerting policy that monitors your usage of Google Cloud Observability and alerts you when you approach the threshold for billing. For more information about this, please refer to the documentation.

[\[https://cloud.google.com/stackdriver/pricing#alert-usage\]](https://cloud.google.com/stackdriver/pricing#alert-usage)

Creating an alerting policy

Create new alerting policy

1 Conditions

Basic Conditions

HTTP check on instance summer01

Violates when: Uptime Check Health on Instance (GCE) summer01 fails

Edit Delete

+ Add Another Condition

2 Notifications (optional)

When alerting policy violations occur, you will be notified via these channels. [Learn more](#)

Email

demo@example.com

X

+ Add Another Notification

3 Documentation (optional)

When email notifications are sent, they'll include any text entered here. This can convey useful information about the problem and ways to approach fixing it.

Edit Preview

Markdown Formatting Help

 Main Server health check failed

+ Server named summer01 failed a Stackdriver uptime check

+ IP Address of the server is: 104.197.58.79

4 Name this policy

A policy's name is used in identifying which policies were triggered, as well as managing configurations of different policies.

Uptime Check Policy

Save Policy




Cancel

Here is an example of what creating an alerting policy looks like. On the left, you can see an HTTP check condition on the summer01 instance. This will send an email that is customized with the content of the documentation section on the right.

Let's discuss some best practices when creating alerts:

- We recommend alerting on symptoms, and not necessarily causes. For example, you want to monitor failing queries of a database and then identify whether the database is down.
- Next, make sure that you are using multiple notification channels, like email and SMS. This helps avoid a single point of failure in your alerting strategy.
- We also recommend customizing your alerts to the audience's needs by describing what actions need to be taken or what resources need to be examined.
- And finally, avoid noise, because this will cause alerts to be dismissed over time. Specifically, adjust monitoring alerts so that they are actionable and don't just set up alerts on everything possible.

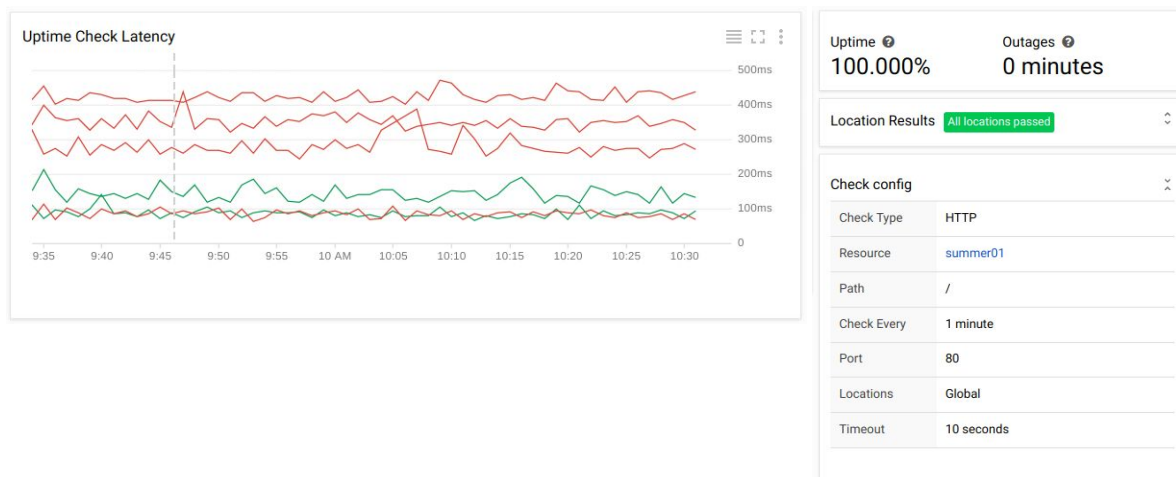
Uptime checks test the availability of your public services

CHECKS	VIRGINIA	OREGON	IOWA	BELGIUM	SINGAPORE	SAO PAULO	POLICIES
Instance 1	✓	✓	✓	✓	✓	✓	
Instance 2	✓	✓	✓	✓	✓	✓	
Instance 3	✓	✓	✓	✓	✓	✓	

Uptime checks can be configured to test the availability of your public services from locations around the world, as you can see on this slide. The type of uptime check can be set to HTTP, HTTPS, or TCP. The resource to be checked can be an App Engine application, a Compute Engine instance, a URL of a host, or an AWS instance or load balancer.

For each uptime check, you can create an alerting policy and view the latency of each global location.

Uptime check example

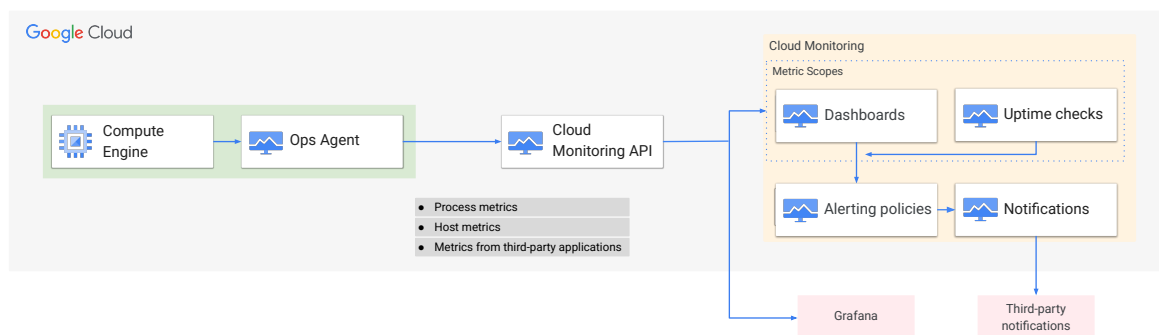


Here is an example of an HTTP uptime check. The resource is checked every minute with a 10-second timeout. Uptime checks that do not get a response within this timeout period are considered failures.

So far there is a 100% uptime with no outages.

What is Ops Agent?

Ops Agent gathers system and application metrics from **VM instances** and sends them to Monitoring.



Google Cloud

Monitoring data can originate at a number of different sources. With Google Compute Engine instances, because the VMs are running on Google hardware, the hypervisor cannot access some of the internal metrics inside a VM, for example, memory usage. The Ops Agent collects metrics inside the VM, not at the hypervisor level.

The Ops Agent is the primary agent for collecting telemetry data from your Compute Engine instances.

This diagram shows how data is collected to monitor workloads running on a Compute Engine instance. Ops Agent installed on Compute Engine collects data beyond the system metrics. The collected metric is then used by Cloud Monitoring to create dashboards, alerts, uptime checks and notifications to drive observability for workloads running in your application.

You can configure the Ops Agent to monitor many third-party applications. For a detailed list, refer to the [documentation](#).

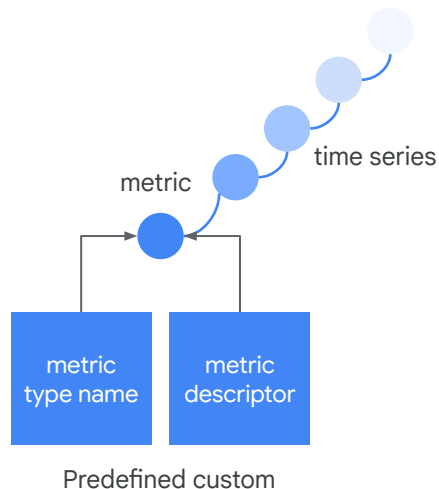
The Ops Agent supports most major operating systems such as CentOS, Ubuntu and Windows.

Custom metrics

Custom metric example in Python:

```
client = monitoring.Client()
descriptor = client.metric_descriptor(
    'custom.googleapis.com/my_metric',

    metric_kind=monitoring.MetricKind.GAUGE,
    value_type=monitoring.ValueType.DOUBLE,
    description='This is a simple example
of a custom metric.')
descriptor.create()
```



Predefined custom

If the standard metrics provided by Cloud Monitoring do not fit your needs, you can create custom metrics.

For example, imagine a game server that has a capacity of 50 users. What metric indicator might you use to trigger scaling events? From an infrastructure perspective, you might consider using CPU load or perhaps network traffic load as values that are somewhat correlated with the number of users. But with a custom metric, you could actually pass the current number of users directly from your application into Cloud Monitoring.

To get started with creating custom metrics, please refer to the documentation.

[\[https://cloud.google.com/monitoring/custom-metrics/creating-metrics#monitoring-create-metric-python\]](https://cloud.google.com/monitoring/custom-metrics/creating-metrics#monitoring-create-metric-python)

Autoscale to maintain a metric at a target value

To maintain a metric at a target value, specify a utilization target.

If the metric comes from each VM in your MIG,



the average metric value across all VMs is compared with the utilization target.

If the metric applies to the whole MIG and does not come from the VMs in your MIG,



the metric value is compared with the utilization target.

If your metric has multiple values,



apply a filter to autoscale using an individual value from the metric.

Google Cloud

When you want to maintain a metric at a target value, specify a utilization target. The autoscaler creates VMs when the metric value is above the target and deletes VMs when the metric value is below the target.

- If the metric comes from each VM in your managed instance group (MIG), then the autoscaler takes the average metric value across all VMs in the MIG and compares it with the utilization target.
- If the metric applies to the whole MIG and does not come from the VMs in your MIG, then the autoscaler compares the metric value with the utilization target.

When your metric has multiple values, apply a filter to autoscale using an individual value from the metric.

Lab Intro

Resource Monitoring



Google Cloud

Let's take some of the monitoring concepts that we just discussed and apply them in a lab.

In this lab, you learn how to use Cloud Monitoring to gain insight into applications that run on Google Cloud. Specifically, you will enable Cloud Monitoring, add charts to dashboards and create alerts, resource groups, and uptime checks.

Lab Review

Resource Monitoring



Google Cloud

In this lab, you got an overview of Cloud Monitoring. You learned how to monitor your project, create alerts with multiple conditions, add charts to dashboards, create resource groups, and create uptime checks for your services.

Monitoring is critical to your application's health, and Cloud Monitoring provides a rich set of features for monitoring your infrastructure, visualizing the monitoring data, and triggering alerts and events for you.

You can stay for a lab walkthrough, but remember that Google Cloud's user interface can change, so your environment might look slightly different.

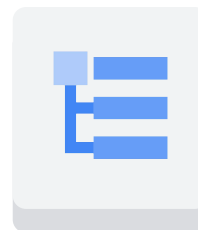


Logging

Monitoring is the basis of Google Cloud Observability, but the service also provides logging, error reporting, and tracing. Let's learn about logging.

Logging

- Platform, systems, and application logs
 - API to write to logs
 - 30-day retention
- Log search/view/filter
- Log-based metrics
- Monitoring alerts can be set on log events
- Data can be exported to Cloud Storage, BigQuery, and Pub/Sub



Logging

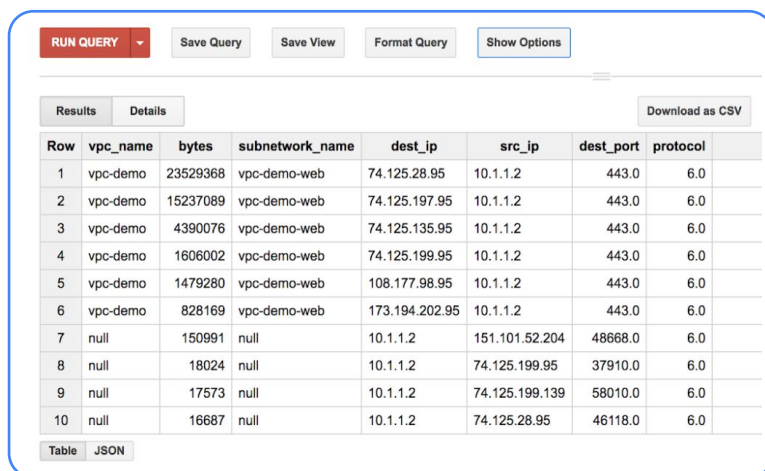
Cloud Logging allows you to store, search, analyze, monitor, and alert on log data and events from Google Cloud and AWS. It is a fully managed service that performs at scale and can ingest application and system log data from thousands of VMs.

Logging includes storage for logs, a user interface called Logs Explorer, and an API to manage logs programmatically. The service lets you read and write log entries, search and filter your logs, and create log-based metrics.

Logs are only retained for 30 days, but you can export your logs to Cloud Storage buckets, BigQuery datasets, and Pub/Sub topics.

Exporting logs to Cloud Storage makes sense for storing logs for more than 30 days, but why should you export to BigQuery or Pub/Sub?

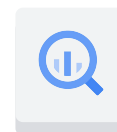
Analyze logs in BigQuery and visualize in Looker Studio



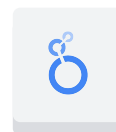
The screenshot shows the BigQuery interface with a query result table. The table has columns: Row, vpc_name, bytes, subnetwork_name, dest_ip, src_ip, dest_port, and protocol. The data is as follows:

Row	vpc_name	bytes	subnetwork_name	dest_ip	src_ip	dest_port	protocol
1	vpc-demo	23529368	vpc-demo-web	74.125.28.95	10.1.1.2	443.0	6.0
2	vpc-demo	15237089	vpc-demo-web	74.125.197.95	10.1.1.2	443.0	6.0
3	vpc-demo	4390076	vpc-demo-web	74.125.135.95	10.1.1.2	443.0	6.0
4	vpc-demo	1606002	vpc-demo-web	74.125.199.95	10.1.1.2	443.0	6.0
5	vpc-demo	1479280	vpc-demo-web	108.177.98.95	10.1.1.2	443.0	6.0
6	vpc-demo	828169	vpc-demo-web	173.194.202.95	10.1.1.2	443.0	6.0
7	null	150991	null	10.1.1.2	151.101.52.204	48668.0	6.0
8	null	18024	null	10.1.1.2	74.125.199.95	37910.0	6.0
9	null	17573	null	10.1.1.2	74.125.199.139	58010.0	6.0
10	null	16687	null	10.1.1.2	74.125.28.95	46118.0	6.0

Below the table, there are tabs for 'Table' and 'JSON', and a 'Download as CSV' button.



BigQuery



Looker Studio

Exporting logs to BigQuery allows you to analyze logs and even visualize them in Looker Studio.

BigQuery runs extremely fast SQL queries on gigabytes to petabytes of data. This allows you to analyze logs, such as your network traffic, so that you can better understand traffic growth to forecast capacity, network usage to optimize network traffic expenses, or network forensics to analyze incidents.

For example, in this screenshot I queried my logs to identify the top IP addresses that have exchanged traffic with my web server. Depending on where these IP addresses are, and who they belong to, I could relocate part of my infrastructure to save on networking costs or deny some of these IP addresses if I don't want them to access my web server.

If you want to visualize your logs, I recommend connecting your BigQuery tables to Looker Studio. Looker Studio transforms your raw data into the metrics and dimensions that you can use to create easy-to-understand reports and dashboards.

I mentioned that you can also export logs to Pub/Sub. This enables you to stream logs to applications or endpoints.



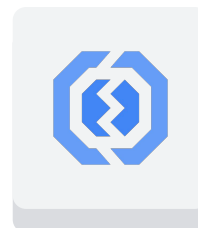
Error Reporting

Let's learn about another feature of Google Cloud Observability: Error Reporting.

Error Reporting

Aggregate and display errors for running cloud services

- Error notifications
- Error dashboard
- App Engine, Apps Script, Compute Engine, Cloud Run, Cloud Run functions, GKE, Amazon EC2
- Go, Java, .NET, Node.js, PHP, Python, and Ruby



Error Reporting

Error Reporting counts, analyzes, and aggregates the errors in your running cloud services. A centralized error management interface displays the results with sorting and filtering capabilities, and you can even set up real-time notifications when new errors are detected.

Currently, Error Reporting is generally available for App Engine on both standard and flexible environments, Apps Script, Compute Engine, Cloud Run, Cloud Run functions, Google Kubernetes Engine, and Amazon EC2.

In terms of programming languages, the exception stack trace parser is able to process Go, Java, .NET, Node.js, PHP, Python, and Ruby.



Tracing

Tracing is another Google Cloud Observability feature integrated into Google Cloud.

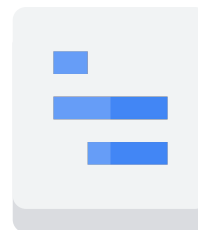
Tracing

Tracing system

- Displays data in near real-time
- Latency reporting
- Per-URL latency sampling

Collects latency data

- App Engine
- Global external Application Load Balancers
- Applications instrumented with the Cloud Trace SDKs



Trace

Cloud Trace is a distributed tracing system that collects latency data from your applications and displays it in the Google Cloud console. You can track how requests propagate through your application and receive detailed near real-time performance insights.

Cloud Trace automatically analyzes all of your application's traces to generate in-depth latency reports that surface performance degradations and can capture traces from App Engine, global external Application Load Balancers, and applications instrumented with the Cloud Trace API.

Managing the amount of time it takes for your application to handle incoming requests and perform operations is an important part of managing overall application performance. Cloud Trace is actually based on the tools used at Google to keep our services running at extreme scale.

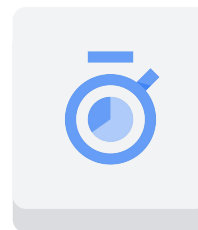


Profiling

Finally, let's cover the last feature of Google Observability in this module, which is the profiler.

Profiling

- Continuously analyze the performance of CPU or memory-intensive functions executed across an application.
- Uses statistical techniques and extremely low-impact instrumentation.
- Runs across all production instances.
- Java, Go, Node.js, and Python



Profiler

Poorly performing code increases the latency and cost of applications and web services every day. Cloud Profiler continuously analyzes the performance of CPU or memory-intensive functions executed across an application.

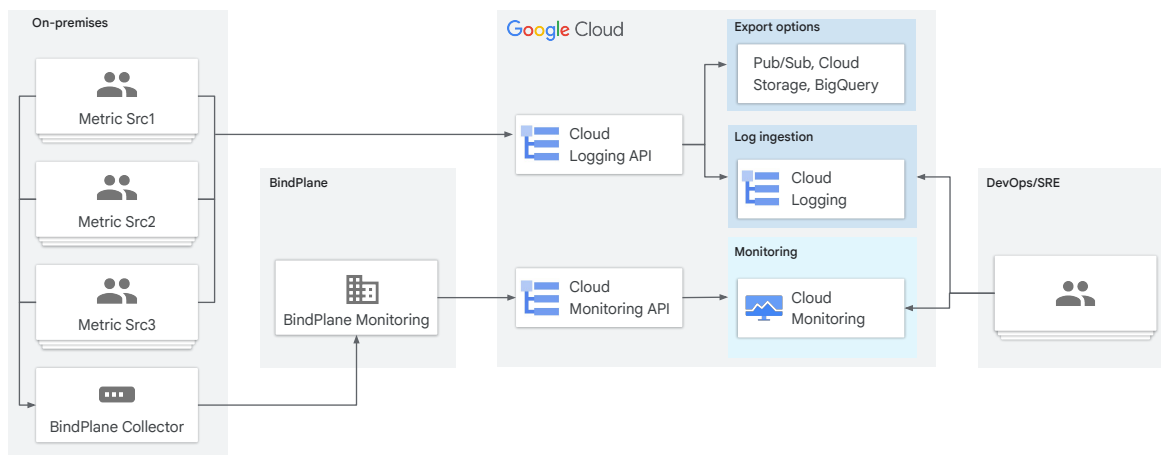
While it's possible to measure code performance in development environments, the results generally don't map well to what's happening in production. Many production profiling techniques either slow down code execution or can only inspect a small subset of a codebase. Profiler uses statistical techniques and extremely low-impact instrumentation that runs across all production application instances to provide a complete picture of an application's performance without slowing it down.

Profiler allows developers to analyze applications running anywhere, including Google Cloud, other cloud platforms, or on-premises, with support for Java, Go, Node.js, and Python.



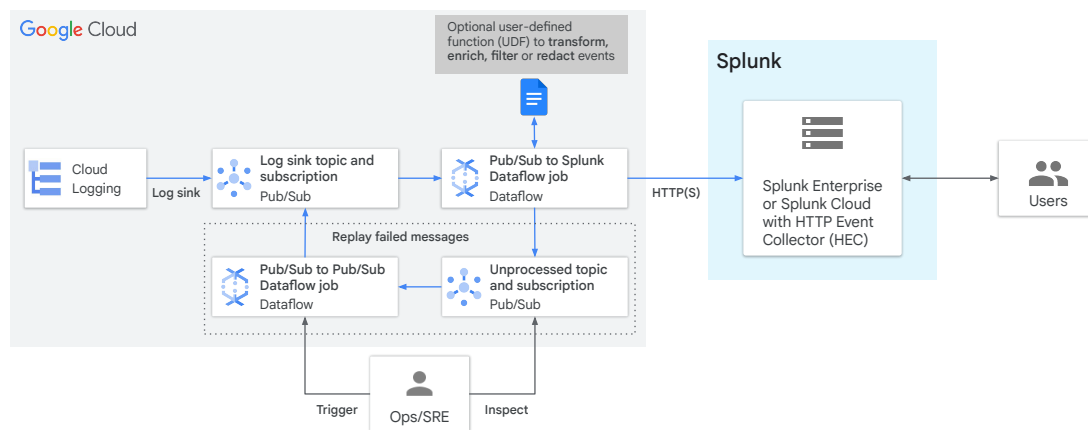
Google Cloud Observability also supports a rich and growing ecosystem of technology partners, as shown on this slide. This helps expand the IT ops, security, and compliance capabilities available to Google Cloud customers.

Reference architecture for logging and monitoring on-premises resources with BindPlane



The diagram shows the architecture of how BindPlane ingests logs and then how those logs are ingested into Cloud Logging plus collects metrics and then how these metrics are ingested into Cloud Monitoring. Adding Blue Medora's software, BindPlane, helps collect the metrics and logs and push them into the open APIs that form our core observability platform. Once a log source is ingested into Cloud Logging, you can view and search through the raw log data and create metrics off of those log files just like logs collected from Google Cloud. You can use all the features of Google Cloud Observability, including viewing logs in real time in the Google Cloud console or through log-based metrics to view logs and metrics side by side and alert on logs.

Reference architecture for log data flows from Google Cloud to Splunk



Google Cloud

This diagram depicts the architecture involved in filtering and exporting log data from Cloud Logging to Splunk Enterprise. Pub/Sub is used to temporarily store logged messages as they are published from Cloud Logging, before delivering them to Splunk.

Cloud Logging begins by gathering logs into a centralized location, then forwards them to Google Cloud's Pub/Sub messaging service. Pub/Sub manages the log data using a dedicated channel. A primary Dataflow pipeline extracts logs from Pub/Sub and delivers them to Splunk for analysis. To safeguard against errors, a secondary Dataflow pipeline runs in parallel, capable of resending logs if delivery fails. Finally, Splunk (deployed on-premises, in Google Cloud as SaaS, or via a hybrid approach) receives the logs and enables in-depth analysis.

The workflow uses a streaming pipeline to natively push your Google Cloud data to your Splunk Cloud or Splunk Enterprise instance using the Pub/Sub to Splunk Dataflow template. Using this Dataflow template, you can export data from Pub/Sub to Splunk. So, any message that can be delivered to a Pub/Sub topic can now be forwarded to Splunk.

For more information about integrations, refer to the link in the Course Resources.

[\[https://cloud.google.com/stackdriver/partners\]](https://cloud.google.com/stackdriver/partners)

Review: Resource Monitoring



In this module, I gave you an overview of Google Cloud Observability and its monitoring, logging, error reporting, fault tracing, and profiling features. Having all of these integrated into Google Cloud allows you to operate and maintain your applications, which is known as site reliability engineering or SRE.

If you're interested in learning more about SRE, you can explore the SRE book or some of our SRE courses.

Review

Essential Cloud Infrastructure:
Core Services



Google Cloud

Thank you for taking the “Essential Cloud Infrastructure: Core Services” course. I hope you have a better understanding of how to administer IAM, choose between the different data storage services in Google Cloud, examine billing of Google Cloud resources and monitor those resources. Hopefully the demos and labs made you feel more comfortable using the different Google Cloud services that we covered.

Elastic Cloud Infrastructure: Scaling and Automation

01	Interconnecting Networks
02	Load Balancing and Autoscaling
03	Infrastructure Automation
04	Managed Services



Next, I recommend enrolling in the “Elastic Cloud Infrastructure: Scaling and Automation” course of the “Architecting with Google Compute Engine” series.

1. In that course, we start by going over the different options to interconnect networks to enable you to connect your infrastructure to Google Cloud.
2. Next, we’ll go over Google Cloud’s load balancing and autoscaling services, which you will get to explore directly.
3. Then, we’ll cover infrastructure automation services like Terraform, so that you can automate the deployment of Google Cloud infrastructure services.
4. Lastly, we’ll talk about other managed services that you might want to leverage in Google Cloud.

Enjoy that course!